IN THE CLAIMS:

The following listing of claims will replace all prior versions, and listings, of claims in the application.

1. – 6. (Canceled)

7. (Currently Amended) ~~An application processing network~~ A system comprising:
at least one processor; and
at least one memory coupled to the at least one processor, wherein the at least one memory is configured to store:
~~an application capable of generating filesystem operations;~~
a vnode layer configured to receive a plurality of requests for filesystem operations ~~from the~~ generated by at least one application, wherein the plurality of requests for the filesystem operations comprise an open() request specifying a file; and
an overlay filesystem ~~including~~ comprising:
a back filesystem containing shared read-only files; and
a front filesystem mounted above the back filesystem and containing writable files[,];
wherein the overlay filesystem ~~being~~ is configured to:
selectively route the plurality of requests for the filesystem operations from the vnode layer to the front and back filesystems; and
allocate an onode in the at least one memory upon opening the file specified by the open() request, wherein the onode corresponds to the file specified by the open() request and comprises a shadow vnode.

8. (Currently Amended) The ~~application processing network~~ system of claim 7, wherein ~~a filesystem operation is an open() request for a file and~~ the overlay filesystem is configured to:

~~allocate an onode featuring a shadow vnode,~~

send the open() request to the front filesystem which returns a front vnode to be stored in the onode, and

send the open() request to the back filesystem which returns a back vnode to be stored in the onode.

9. (Currently Amended) The ~~application processing network~~ system of claim 8, wherein the shadow vnode maintains a reference count that is incremented each time the file is opened.

10. (Currently Amended) The ~~application processing network~~ system of claim 8, wherein the ~~onende~~ onode and the shadow vnode are linked and the shadow vnode is returned to the vnode layer.

11. (Currently Amended) The ~~application processing network~~ system of claim 10, wherein the vnode layer returns to the application a file descriptor linked to the shadow vnode.

12. (Currently Amended) The ~~application processing network~~ system of claim 7, wherein the plurality of requests for the filesystem operations comprise a read() request specifying a second file;

wherein ~~a filesystem operation is a read() request for a file and~~ the overlay filesystem is configured to:

receive the read() request and a second shadow vnode from the vnode layer[,];

determine a vnode for a filesystem from an onode for the second file[,]; and

pass the read() request to the filesystem.

13. (Currently Amended) The ~~application processing network~~ system of claim 12, wherein the vnode for the filesystem is a front vnode for the front filesystem.

14. (Currently Amended) The ~~application processing network~~ system of claim 7, wherein a structure of the vnode layer is filesystem-independent; and wherein the overlay filesystem maintains a plurality of onodes.

15. (Currently Amended) The ~~application processing network~~ system of claim 14, wherein each onode ~~can include~~ comprises a vnode triplet.

16. (Currently Amended) The ~~application processing network~~ system of claim 15, wherein the vnode triplet includes a shadow vnode pointer, a front vnode pointer, and a back vnode pointer that point to a shadow vnode, a front vnode, and a back vnode, respectively.

17. (Currently Amended) The ~~application processing network~~ system of claim 14, wherein each onode is stored in a hash table.

18. (Currently Amended) The ~~application processing network~~ system of claim 7, wherein the overlay filesystem includes more than two filesystems and is further configured to allocate and cache ~~onondes~~ a plurality of onodes.

19. (Currently Amended) The ~~application processing network~~ system of claim 7, wherein the overlay filesystem is further configured to support a snapshot/restore module.

20. (Currently Amended) The ~~application processing network~~ system of claim 7, wherein the overlay filesystem is further configured to implement a file in the front filesystem with a page-level copy-on-write structure.

21. (Currently Amended) The ~~application processing network~~ system of claim 20, wherein a format of the file in the front filesystem includes a header, a page map, and a page.

22. (Currently Amended) The ~~application processing network~~ system of claim 21, wherein the header stores extended file attributes, file verification data, virtual size, and reserved/padding.

23.. (Currently Amended) The ~~application processing network~~ system of claim 21, wherein the page map ~~includes~~ comprises a bitmap indicating a location of the page in the front filesystem.

24. (New) A computer-implemented method comprising:
receiving a plurality of requests for filesystem operations at a vnode layer, wherein the plurality of requests for filesystem operations are generated by at least one application, and wherein the plurality of requests for the filesystem operations comprise an open() request specifying a file;
selectively routing the plurality of requests for the filesystem operations from the vnode layer to a back filesystem and a front filesystem in an overlay filesystem, wherein the back filesystem contains shared read-only files, and wherein the front filesystem is mounted above the back filesystem and contains writable files; and
allocating an onode in memory upon opening the file specified by the open() request, wherein the onode corresponds to the file specified by the open() request and comprises a shadow vnode.

25. (New) A computer-readable medium comprising program instructions, wherein the program instructions are computer-executable to perform:
receiving a plurality of requests for filesystem operations at a vnode layer, wherein the plurality of requests for filesystem operations are generated by at least one application, and wherein the plurality of requests for the filesystem operations comprise an open() request specifying a file;
selectively routing the plurality of requests for the filesystem operations from the vnode layer to a back filesystem and a front filesystem in an overlay

filesystem, wherein the back filesystem contains shared read-only files, and wherein the front filesystem is mounted above the back filesystem and contains writable files; and

allocating an onode in memory upon opening the file specified by the open() request, wherein the onode corresponds to the file specified by the open() request and comprises a shadow vnode.